IN THE SPECIFICATION:

Please replace the first paragraph on Page 1, with the following amended paragraph:

This application claims priority to European Application Serial No. 00402331.3, filed August 21, 2000 (TI-31366EU) and to European Application Serial No. 01400687.8, filed March 15, 2001 (TI-31354EU). US Patent Application Serial No. [_____] 09/932,651 (TI-31366US) is incorporated herein by reference.

Please replace paragraph [28] with the following amended paragraph:

[28] The hit-miss or hit-hit signal has precedence over the hit way (2-5) signals ~~524~~ 514 of the 4-way set-associative cache. This implies that any value loaded previously in the cache that should be in the RAM-set is never selected and will eventually be removed from the cache. However, data can create coherency problem in case of modified data (copy back). Therefore, it is recommended to write back ("clean") or even flush the range of address that will correspond to the RAM-set range of addresses. Other embodiments might not have such precedence defined and instead rely on cache invalidate operations to correctly prepare an address range that will be programmed to reside in a RAM-set, for example.

Please replace paragraph [80] with the following amended paragraph:

[80] A clean and flush operation is also performed using the same flow. In this case[[.]], in steps 1204 and 1214, the selected valid bit(s) and the selected dirty bit(s) are checked. If a valid bit is in a valid state, it reset to an invalid state in steps 1206, 1216. If a dirty bit is asserted, the associated segment is written to secondary memory in step 1206, 1216.

Please replace Table 2, on page 23 with the following amended table:

Table 1 – Cache and RAM Control Operations

(C: operation on the cache, RS: operation on RAM-set, R: operation on RAM)

Function		Software view (memory mapped/ co-proc)
Flush_entry (address)	C/RS	Flush the ~~entry1~~,entry whose address matches the provided address or a Range of addresses, if End has been set previously. Flush-range instruction is made of two consecutive instructions Set_End_addr(address) + Flush_entry (address).
Flush_all_entry_of_ task_ID(task_ID)	C	Flush all entries matching to the current taskID in the cache but not in the RAM-set
Flush_all_entry_of_R_ID(task_ID)	C	Flush all entries matching to the current R_ID in the cache but not in the RAM-set
Flush_all	C	Flush all entries in the cache but not in RAM-set
Flush_all_shared	C	Flush all entries marked as shared
Flush_all_task_ID_ shared(task_ID)	C	Flush all entries matching the current taskID and marked as shared
Flush_all_task_ID_not_ shared (task_ID)	C	Flush all entries matching the current taskID and marked as not shared
Clean_entry (address)	C/RS	Clean the ~~entry1~~,entry, whose address matches the provided address or a Range of address if End has been set previously. Clean-range instruction is made of two consecutive instructions Set_End_addr(address) + Clean_entry (address).
Clean_all_entry_of_ taskID(task_ID)	C	Clean all entries matching to the current taskID in the cache but not in the RAM-set
Clean_all_entry_0f_R_ ID(task_ID)	C	Clean all entries matching to the current R_ID in the cache but not in the RAM-set
Clean_all	C	Clean all entries in the cache but not in RAM-set
Clean_all_shared	C	Clean entries marked as shared
Flush_all_task_ID_ shared(task_ID)	C	Flush all entries matching the current taskID and marked as shared
Clean_all_taskID_not_ shared(Task_ID)	C	Clean all entries matching the current taskID and marked as not shared
Clean&Flush_single_ entry(address)	C/RS	Clean and flush the ~~entry1~~,entry, whose address matches the provided address or a Range of address if End has been set previously. Clean-range instruction is made of two consecutive instructions Set_End_addr(address) + Clean_entry (address).
Clean&flush_all_ entry_of_taskID (Task_ID)	C	Clean and flush all entries matching to the current taskID in the cache but not in the RAM-set
Clean&flush_all_entry_ of_R_ID (Task_ID)	C	Clean and flush all entries matching to the current R_ID in the cache but not in the RAM-set
Clean&flush_all	C	Clean and flush all entries in the cache but not in RAM-set
Clean&flush_all_ shared	C	Clean and flush entries marked as shared
Clean&flush_all_ taskID_shared (task_ID)	C	Clean and flush all entries matching the current taskID and marked as shared
Clean&flush_all_taskID_ not_shared (task_ID)	C	Clean and flush all entries matching the current taskID and marked as not shared
Set_RAM_Set_Base_ addr(RAM-setID)	RS/R	Set new RAM-set base address, set VG and clear all VI and set End to last RAM-set address by default preparing the full RAM-set loading. In that case no need to write the END address before writing the start address to load the RAM-set
Set_End_Addr (address)	C/RS	Set end address of the next block load and set the RAM-set controller in block fill mode.
Set_start_addr (address)	C/RS	Set start address of a block and initiates the loading of this block
Prefetch-entry(address	C/RS	Prefetch-the entry, whose address matches the provided address or a Range of address if End has been set previously. Prefetch-range instruction is made of two consecutive instructions Set_End_addr(address) + Prefetch_entry (address).
Flush_RAM-set (RAMset_ID)	RS/R	Clear VG and all VI of the selected RAM-set